ENERGETIC MATERIAL SIMULATIONS: ADVANCING THE FUTURE FORCE

William D. Mattson* and Betsy M. Rice
U. S. Army Research Laboratory, AMSRD-ARL-WM-BD
Aberdeen Proving Ground, MD 21005-5069

ABSTRACT

We describe a new extensible software system to perform molecular simulations of energetic materials. A new approach for extensible software development based on an XML description of a program structure and a set of components stored in shared object libraries is described first. A specific example of molecular dynamics simulations for energetic materials is given next and this is finally expanded with the capability to perform molecular packing calculations to show the extensibility and applicability of the system.

1. INTRODUCTION

Advanced Energetic Materials (AEM) represents a technology area of enormous importance to the DOD. AEM are required to enable high priority military missions ranging from Hard and Deeply Buried Target Defeat, to Advanced Propulsion, to lightened highly mobile force evolution and the thrust towards miniaturized munitions and systems. It is recognized that weapons superiority is dependent on the development of AEM. Unfortunately, the current national AEM investment is sub-critical. It is the consensus of the US technology community that energetics technology is an area where we have been surprised by foreign achievements in the past and, given the vastly larger investment in these key technologies by foreign nations, are highly vulnerable in the field in the future. Overseen by the Office of the Under Secretary of Defense (Science and Technology) and the Office of Munitions, the National Advanced Energetics Initiative (NAEI) has been charted. The NAEI recognizes that developments in computational chemistry and physics-based modeling using High-Performance Computing, chemical synthesis and formulation, and materials science are providing the key factors that will provide breakthroughs in the performance of energetic materials. The DOD HPC network and advanced modeling science and technologies afford a critical means to rapidly close the technology gap and expedite the design and development of new revolutionary AEM.

Our contribution to the design and development of AEM has been to establish a computational framework that will allow easy integration of evolving software required to support the modeling needs of the Army, with rapid turnaround. The paper describes a generative programming approach [1] to produce a suite of efficient, user-friendly, highly scalable molecular simulation codes to study reactive and non-reactive processes in energetic materials. The core of this approach is a simulation generator that assembles and runs simulations described in eXtensible Markup Language (XML) from a set of components in shared object libraries. The complete set of standard molecular simulation components can be combined in any fashion creating typical simulations and providing unanticipated functionality. This flexible software can be extended without modifying or recompiling the existing code by adding shared object libraries with the new desired functionality. This extensibility allows the code to adapt to the changing needs within the Army while providing the world class computational performance needed for energetic materials research, creating a focal point for the integration of emerging science and high performance computing.

The first software suite integrated into this package, which was supported by the DOD High Performance Computing Modernization Program CHSSI project CCM-5, allows multi-million atom molecular dynamics simulations in a variety of thermodynamic ensembles, including the recently developed uniaxial Hugoniostat method [2].

The next extension of the package is the integration of "ab initio crystal prediction" software, supported under CHSSI MBD-4. This procedure predicts the crystal structure and density of a solid using only the molecular structure of a single molecule. This predictive capability is considered crucial to the design and development process of AEM, since one of the fundamental properties required for the initial screening of a candidate energetic material is its crystalline density. This density provides estimates of idealized performance in a gun and its detonation velocity and pressure. Integration of this computational capability into this extensible framework will result in the capability to predict properties associated with performance of an energetic material within a time span of days, making it a critical screening tool for candidate materials. The traditional descriptions of the inter-atomic forces in this method are empirical in nature, thus limiting the predictive capability. To overcome this obstacle, we are developing an interface to incorporate solid-state quantum mechanical software packages, specifically DOD Planewave [3] (supported through CHSSI CCM-1). An added benefit to this approach is

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
00 DEC 2004	**N/A**	**-**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Energetic Material Simulations: Advancing The Future Force	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **U. S. Army Research Laboratory, AMSRD-ARL-WM-BD Aberdeen Proving Ground, MD 21005-5069**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release, distribution unlimited

13. SUPPLEMENTARY NOTES
See also ADM001736, Proceedings for the Army Science Conference (24th) Held on 29 November - 2 December 2005 in Orlando, Florida.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**UU**	**5**	

that the integration of the quantum mechanical descriptions of forces within DOD Planewave can be easily extended to use in the existing molecular dynamics component of the package.

2. THE SOFTWARE PARADIGM AND ENVIRONMENT

The overall goal of this project is to develop a suite of efficient, easy-to-use, extendible, portable and scalable molecular simulation tools for simulations of materials of critical DoD interest. To create an easy to use and extensible environment, this suite of software, entitled VisualXMD, relies on graphical user interfaces, and a standards based execution environment to require only a minimal knowledge of the details of the underlying software. At the same time the execution environment is rich enough to allow the expert to create unique advanced simulations.

The execution environment consists of three parts. The first is the language for the description of the simulations called VisualML. This language is used to specify the order and hierarchy of the functional components of the simulation. It can be used to specify a list of functional components to call in order as well as specify what functional components are called by other functional components. The second part is the graphical user interface that allows the user to visually create the VisualML describing a specific simulation. The final part of the execution environment is a program that reads in the VisualML describing a simulation and calls the functional components in the proper sequence. This "Code Executive" also provides an interface to any simulation parameters contained in the VisualML.

2.1. VisualML

VisualML is a simple procedure-based program structure language that follows the rules of XML and is an XML. It does not include any molecular simulation specific statements and is a general-purpose language to describe program structure; therefore it can be reused in any other visual system requiring a program structure language. It contains a minimal set of constructs for describing a simulation's construction. These include three construct: the function list, the function, and the parameter. The function list construct is a list of functional components to be executed in sequential order. The function construct corresponds to a functional component; it must be defined in a VisualML library description described below. Functions can contain other functions as sub-functions. Sub-functions correspond to functional components executed from within the function corresponding to the parent function statement. Sub-functions are listed in the order in which the parent function references them with in the actual source code. The number of sub-functions must correspond to that in the VisualML library description. Each function may have a zero or more parameter statements. The parameter statement has a named attribute 'type' given in the VisualML library description corresponding to a data type. The parameter statement also has a named attribute 'name'. The name is the name of the specific instantiation of the data type and is used by the source to request the value from the Code Executive. The order and of parameters and their data types must correspond to that in the VisualML library description for a statement.

In addition the VisualML can be used to describe the functional components contained in a shared object library. The library description is used to translate the VisualML calls into the entry points into the shared object library entry points. In addition it contains specific functional component information such as the simulation parameters that needs to be specified, and the number and type of functional components that can be called by the specified functional component. This description may be replaced or expanded to correspond to any update or new functionality with modifying any other tools or functional components. This provides maximum extensibility and reusability. With a modified description the Code Executive will be able to use additional modules in new or added to existing shared object libraries, whatever they may be.

2.2. The Code Executive

The Code Executive will read in a VisualML simulation description file and convert it into a parse tree. The Code Executive executes the functional components in the statement list at the top of the parse tree. The parse sub-tree below that function is passed to the functional component. The module may call the Code Executive and pass it the parse sub-tree and the number of the sub-function to execute (This is the method for executing sub-statements, or calling user definable functional components from with in a functional component). The parse tree contains a parameter list for each function with pointers to the memory of the corresponding instantiation data type to be passed to the modules. The interpreter maintains an array of pointers to point to each instantiation of each data type. It is these pointers that are pointed to in the parameter list of the parse tree. All modules must check to see that the data types that they are using have been allocated. The Code Executive is also reusable in that it is independent on a specific set of modules.

2.3. The Visual Simulator Development Environment

The Visual Simulator Development Environment (VSDE) uses the library description, so it can be used to visually build a simulation out of whatever modules are in the various library descriptions. The VSDE is a graphical

editor for VisualML files. It does not have to be running on the same platform as the Code Executive. The VSDE has two interfaces, the novice interface and the advanced interface. The novice VSDE displays the basic structure of a program and shows the modules that the novice user may change. At a specific changeable module, the GUI will give the user the opportunity to select, from several preset modules, the module that will be used at that point in the simulation. The advanced VSDE has a set of menus containing all the functional components in the library descriptions. The main window contains the simulator window in which the functional components are displayed in their hierarchical order. A right click on the function will display the parameters with their names, and any data values that can be requested by the functional components. The VSDE will get the information about parameters and data from the library descriptions.

3. THE MOLECULAR DYNAMICS FUNCTIONAL COMPONENTS

All module components will be constructed to meet the standards enforced by the parser and interpreter. The modules can then be reused in any VisualML simulator, not just VisualXMD. Generic modules such as program control, domain decomposition, integrators, and visualization tools can be used in construction of a variety of Visual languages. The modules may need modification in order that they can be used with a system that is not based on VisualML.

VisualXMD will be based on a library of routines developed under CHSSI project: CCM-4. These libraries will provide the parallelization and domain decomposition for multiple dimensions. This library consists of approximately 8,000 lines of code. VisualXMD will also leverage the utility of Interdisciplinary Computing Environment (ICE, formerly DICE), which is being developed under several CHSSI programs.

The software suite, using a variety of potentials from the simple to the chemically realistic, will provide a broad range of capabilities and execution options that will enable the effective and easy use of the tools by both the novice and advanced researcher in molecular simulations. and thoroughly tested modules

3.1. Functional Components

Functional Components need to have a C interface to handle the casting of pointers from the parse tree to the call parameter list. The scientific modules will be written in FORTRAN 90 and the C interface and the library description entries will be created and used for all of the FORTRAN 90 modules. Now we will discuss the different classes of modules to be implemented.

3.1.1. Control Components

There are only two types of control modules planned: the conditional module and the loop module. The conditional module will have two sub-statements. The first sub-statement will be executed if the specified conditions are met, otherwise, the second will be executed. Loop modules will have only one sub-statement, which can be a statement list. The loop modules will iterate until the specified conditions are met.

3.1.2. Atom and Bulk Property Control Components

These modules will control specific properties of the simulation. The bulk properties control modules will control boundary conditions and simulation constants. There will be a module that will handle a variety of boundary conditions, based on the previous work on cell-linked lists in CCM-4. These modules will be included in the domain decomposition modules for ease of use.

3.1.3. Integrator Components

The integrator components will all integrate the equations of motion, but they will have different orders and some different properties. The Integrator modules will include, at a minimum, a velocity Verlet integrator and a higher-order integrator, such as an N order Adams-Moulton Predictor-Corrector or Runge-Kutta 4^{th} Order integrator. Recently a component for a uniaxial hugoniostat was developed. This method utilizes modified equations of motion that constrain the system to states that correspond to points on the shock Hugoniot curve. Figure 1 provides a comparison of the shock Hugoniot of a system evaluated using the Hugoniostat method with that obtained from brute force non-equilibrium molecular dynamics (NEMD) simulations of a shocked system of Lennard-Jones particles. The NEMD simulations require extensive computational resources (cycles, processors and memory) in order to accommodate the system size of the simulation; the Hugoniostat method requires only a fraction of the computational resources as the NEMD simulations

3.1.4. Domain Decomposition and Communication Components

The domain decomposition components divide the system up into cells and assign the cells to processors. There is a uniform cell size domain decomposition module. The cell decomposer will divide the system up into multiple cells corresponding to a modified cell linked-list described in W. Mattson and B. M. Rice, *Computer Physics Communications* 119,135 (1999). The domain decomposition component will create or modify a communication structure for use by the communication modules.

3.1.5. Potential Components

Several simple pair-additive and many-body interaction potentials, including the Lennard-Jones, the exponential-six, and the reactive Empirical Bond Order (REBO) potentials for model energetic materials have been implemented.

3.1.6. Property Monitoring Components

These components calculate specific properties of the simulation or atoms, such as Kinetic Energy, Potential Energy, Total Energy, Pressure, Temperature, Volume, and Density.

3.2. Crystal Builder

The crystal builder consists of separate program that generates Cartesian coordinates of a crystalline lattice using user-specific space-group symmetries and fractional coordinates of a molecule. The crystal builder has a graphical user interface.

4. MOLECULAR PACKING

There are three main software components within the scope of this project. Two are extensions from earlier CHSSI CCM projects (DoD Planewave [CCM-1] and VXMD [CCM-5]). The source code for the third package, known as MOLPAK/WMIN, is a mix of non-ANSI-standard Fortran 66 and Fortran 77 and currently exists only in a serial version. Also, current versions of MOLPAK/WMIN are not portable across computational platforms.

The MOLPAK (MOLecular PAcKing) computer program was designed to predict possible crystal structures for small organic compounds. The basic assumption is that of closest packing and MOLPAK creates unit cells that have the smallest volumes per molecule. One can think about a crystal structure in terms of a central molecule surrounded by a coordination sphere of other molecules related to the center by crystallographic symmetry (space group) and the dimensions of the unit cell. Currently, 29 coordination geometries are used for a typical structure search and code has been written for a total of 60 geometries. For each of the coordination geometries, MOLPAK generates from 7000-51000 hypothetical crystal structures from the coordinates of a single molecule. The 25-500 most-dense initial crystal structures then are subjected to lattice energy minimization that optimizes the crystal unit cell parameters and orientation and position of the molecule within the unit cell. The energy minimization is space-group symmetry restricted and performed with the computer code WMIN, which models molecules and crystals in terms of potential energy functions. The refined structures are ranked according to energy with the lowest energy structure taken as the best. In some cases, a combination of density and energy are used to select the best structure and a procedure that involves intermolecular contacts is under development. MOLPAK has been under development for approximately fifteen years under the direction of Professor H. L. Ammon, University of Maryland, and is currently maintained by Professor Ammon. WMIN was written by William R. Busing of Oak Ridge National Laboratory, and published in 1981. To our knowledge, WMIN has not been enhanced nor maintained by the Oak Ridge National Laboratory.

DoD Planewave has its roots in a simple planewave pseudopotential program written in 1992 at NRL to test new ideas about the transformation of pseudopotentials between different representations [Singh, et al. 1992; Singh et al. 1993]. The code was extended and generalized at NRL to test non-local density functionals during the two years [Singh, 1993]. This was in support of a Navy interest in ferroelectric perovskite materials for SONAR applications. Parallelization of the code, and extensions to make it general in terms of symmetry and chemical composition as well as to add forces and dynamics were started in 1996 with support from the CHSSI program. DoD Planewave was the main component of the CCM-1 (the other was ACRES). The parallelization, generalization and documentation of DoD Planewave was done between 1996 and 2000.

The software will allow *ab initio* crystal prediction with quantum mechanical descriptions of forces to obtain reliable densities and structures of materials of interest to the DoD, including energetic materials. This software is an important tool needed to unlock the fundamentals required to relate crystal structure at the lattice level to onset of initiation. This is of high importance to DoD in light of the increasingly complex and harsh environments in which energetic materials will have to perform in current and future systems.

The crystal prediction software has two levels of parallelism. The first and higher level addresses the most computationally intensive portion of the problem: The adequate sampling of orientation space through the generation of crystals based on the orientation of a single molecule. This level of parallelism is a simple replication method where each processor group evaluates a single configuration at a time. In the simple replication method, each processor group communicates with the other processor groups to adaptively partition the remaining configurations, and to report the results of finished configuration calculations. Initially, when there are many configurations remaining to be evaluated, processor groups will usually consist of a single processor and adaptively merge into groups of multiple processors when there are more processors than remaining configurations. The replication method itself is simple to implement, but

the adaptive nature of both the partition of remaining configurations and the processor groups increases the HPC complexity for implementation. The second and lower lever of parallelism is the decomposition of the molecular mechanics for a single configuration across multiple processors. The classical potential and molecular mechanics/energy minimization steps of MOLPAK/ROTPAK and WMIN will be parallelized by domain decomposition within the VXMD framework (CHSSI CCM-5), increasing the capabilities of both VXMD and MOLPAK by making available all permutations of there current capabilities to users. Parallelization will be accomplished through explicit standard message passing interfaces to MPI. DoD Planewave is already parallelized over energy bands for a single configuration but must be interfaced with MOLPAK through VXMD to run simultaneous multiple configurations.

5. CONCLUSIONS

The VisualML framework provides a simple mechanism for assembling new and complex simulations from standard components. The system is capable of a wide variety of efficient and highly scalable simulations necessary for energetics materials research using the components developed under the CHSSI program. It is also easily adaptable to new the technologies being developed for more accurate and complete calculations of the properties of energetic materials.

REFERENCES

[1] K.Czarnecki, and U. Eisenecker, “Generative Programming”, Addison-Wesley (2000).

[2] J.-B. Maillet, M. Mareschal, L. Soulard, R. Ravelo, P. S. Lomdahl, T. C. Germann and B. L. Holian, Phys. Rev. E **63**, 016121 (2000).

[3] See online documentation at http://cst-www.nrl.navy.mil/people/singh/planewave/v3.0/

[4] See online documentation at http://www.arl.hpc.mil/ice/

[5] See the listing under National Center for Supercomputing Applications, "HDF5 - A New Generation of HDF", http://hdf.nsca.uiuc.edu/HDF5

[6] See online documentation at http://xml.apache.org/ and http://www.xml.org

[7] Snir, Otto, Huss-Lederman, Walker, and Dongarra, “MPI-The Complete Reference”, The MIT Press (1998).